DEPARMENT OF INFORMATION AND COMMUNICATION ENGINEERING

TP 13

CRUD APIS, MONGOOSE, ADMIN DASHBOARD

INTERNET PROGRAMMING

Submitted to: lecturer HOK TIN

e20190146
Chhorn Kakada
GIC-A

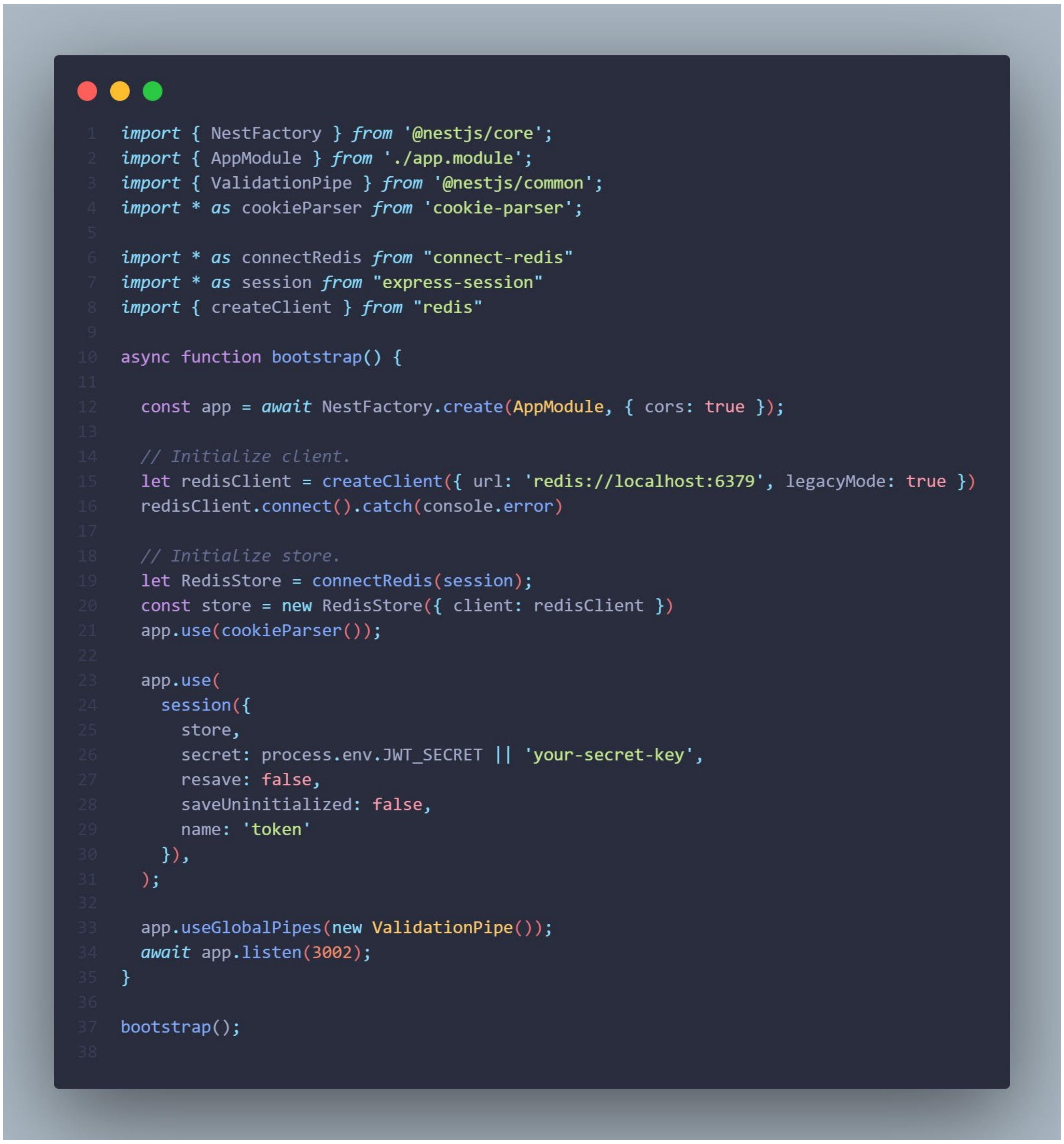
Wed. 05. July. 2023. 11:11PM

Github Account:
https://github.com/ChhornKakada/Intenet_Programming/tree/tp13

Noted: I only done with API.

Implementation of using Redis

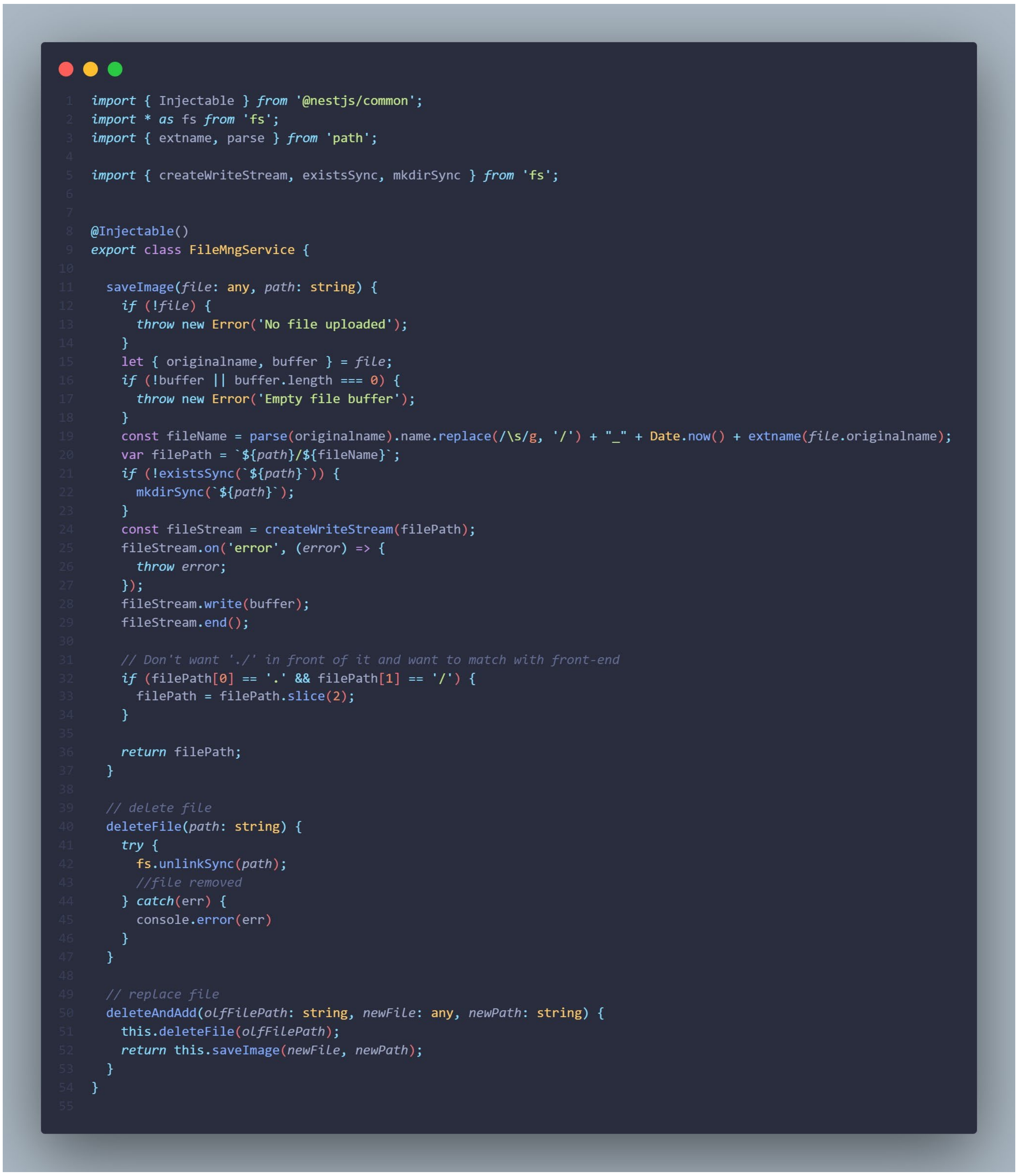
This is how I implement Redis.



```
import { NestFactory } from '@nestjs/core';
import { AppModule } from './app.module';
import { ValidationPipe } from '@nestjs/common';
import * as cookieParser from 'cookie-parser';

import * as connectRedis from "connect-redis"
import * as session from "express-session"
import { createClient } from "redis"

async function bootstrap() {

  const app = await NestFactory.create(AppModule, { cors: true });

  // Initialize client.
  let redisClient = createClient({ url: 'redis://localhost:6379', legacyMode: true })
  redisClient.connect().catch(console.error)

  // Initialize store.
  let RedisStore = connectRedis(session);
  const store = new RedisStore({ client: redisClient })
  app.use(cookieParser());

  app.use(
    session({
      store,
      secret: process.env.JWT_SECRET || 'your-secret-key',
      resave: false,
      saveUninitialized: false,
      name: 'token'
    }),
  );

  app.useGlobalPipes(new ValidationPipe());
  await app.listen(3002);
}

bootstrap();
```

File management

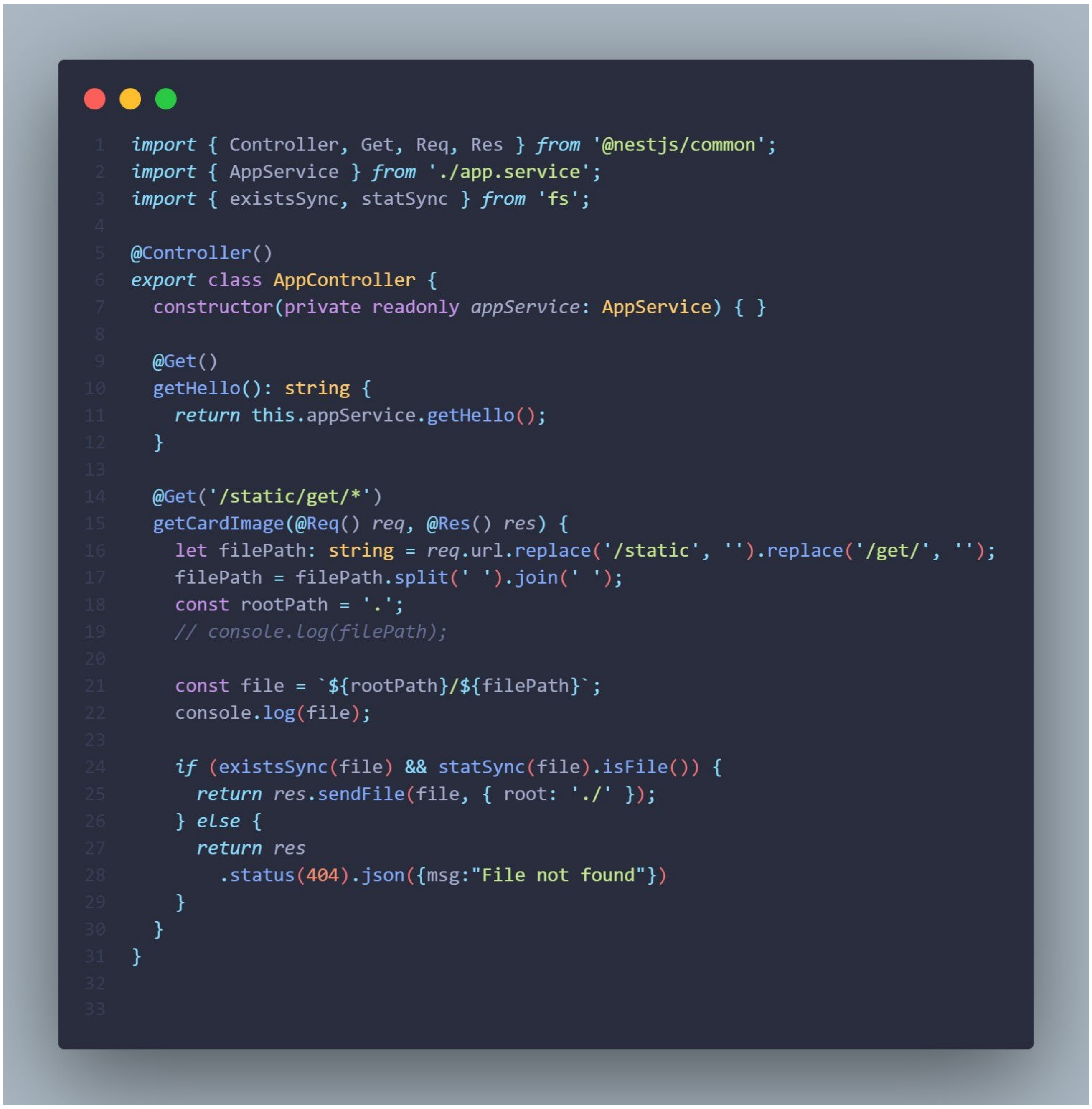
This is the class I implement for **File Management**.



```
import { Injectable } from '@nestjs/common';
import * as fs from 'fs';
import { extname, parse } from 'path';

import { createWriteStream, existsSync, mkdirSync } from 'fs';


@Injectable()
export class FileMngService {

  saveImage(file: any, path: string) {
    if (!file) {
      throw new Error('No file uploaded');
    }
    let { originalname, buffer } = file;
    if (!buffer || buffer.length === 0) {
      throw new Error('Empty file buffer');
    }
    const fileName = parse(originalname).name.replace(/\s/g, '/') + "_" + Date.now() + extname(file.originalname);
    var filePath = `${path}/${fileName}`;
    if (!existsSync(`${path}`)) {
      mkdirSync(`${path}`);
    }
    const fileStream = createWriteStream(filePath);
    fileStream.on('error', (error) => {
      throw error;
    });
    fileStream.write(buffer);
    fileStream.end();

    // Don't want './' in front of it and want to match with front-end
    if (filePath[0] == '.' && filePath[1] == '/') {
      filePath = filePath.slice(2);
    }

    return filePath;
  }

  // delete file
  deleteFile(path: string) {
    try {
      fs.unlinkSync(path);
      //file removed
    } catch(err) {
      console.error(err)
    }
  }

  // replace file
  deleteAndAdd(olfFilePath: string, newFile: any, newPath: string) {
    this.deleteFile(olfFilePath);
    return this.saveImage(newFile, newPath);
  }
}

```

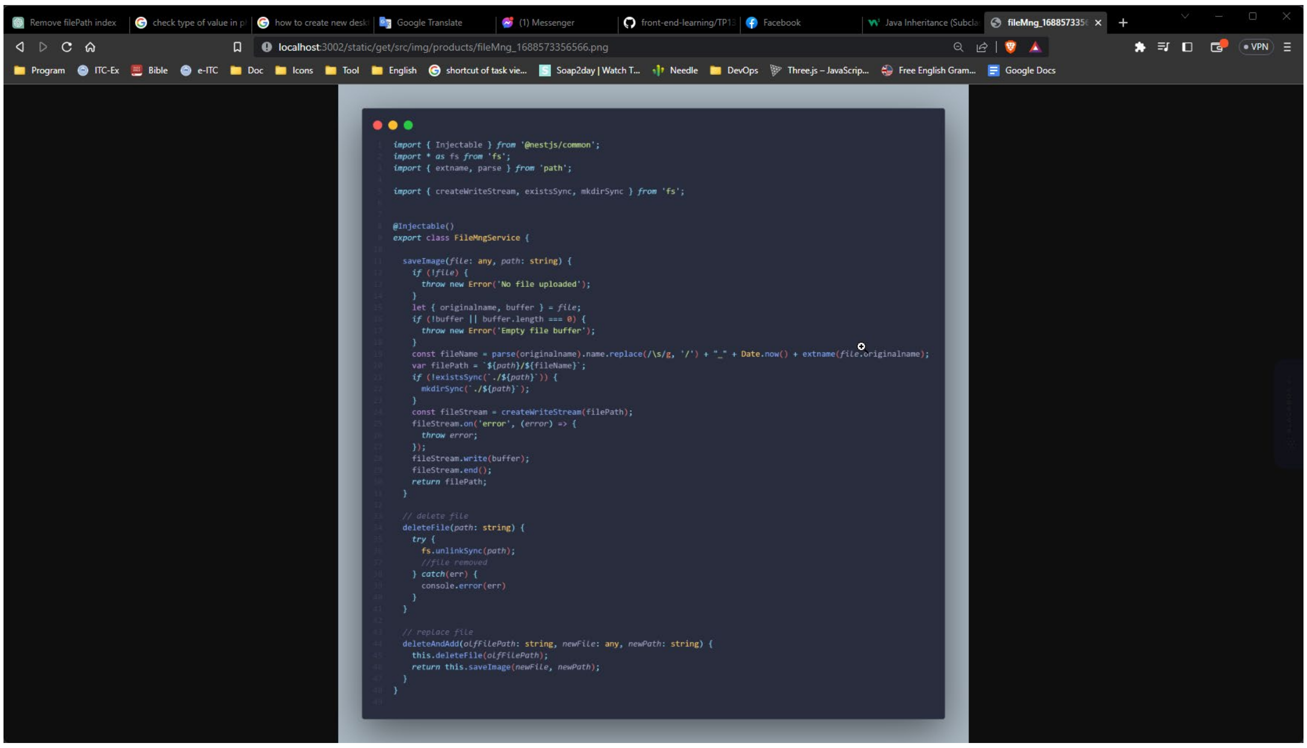
How to access picture in the Url

To access any picture in the Url, we need to modify some code in the **app.controller.ts** to enable user to access it.



```typescript
import { Controller, Get, Req, Res } from '@nestjs/common';
import { AppService } from './app.service';
import { existsSync, statSync } from 'fs';

@Controller()
export class AppController {
  constructor(private readonly appService: AppService) { }

  @Get()
  getHello(): string {
    return this.appService.getHello();
  }

  @Get('/static/get/*')
  getCardImage(@Req() req, @Res() res) {
    let filePath: string = req.url.replace('/static', '').replace('/get/', '');
    filePath = filePath.split(' ').join(' ');
    const rootPath = '.';
    // console.log(filePath);

    const file = `${rootPath}/${filePath}`;
    console.log(file);

    if (existsSync(file) && statSync(file).isFile()) {
      return res.sendFile(file, { root: './' });
    } else {
      return res
        .status(404).json({msg:"File not found"})
    }
  }
}
```

You can access the picture by follow the structure of the image below:

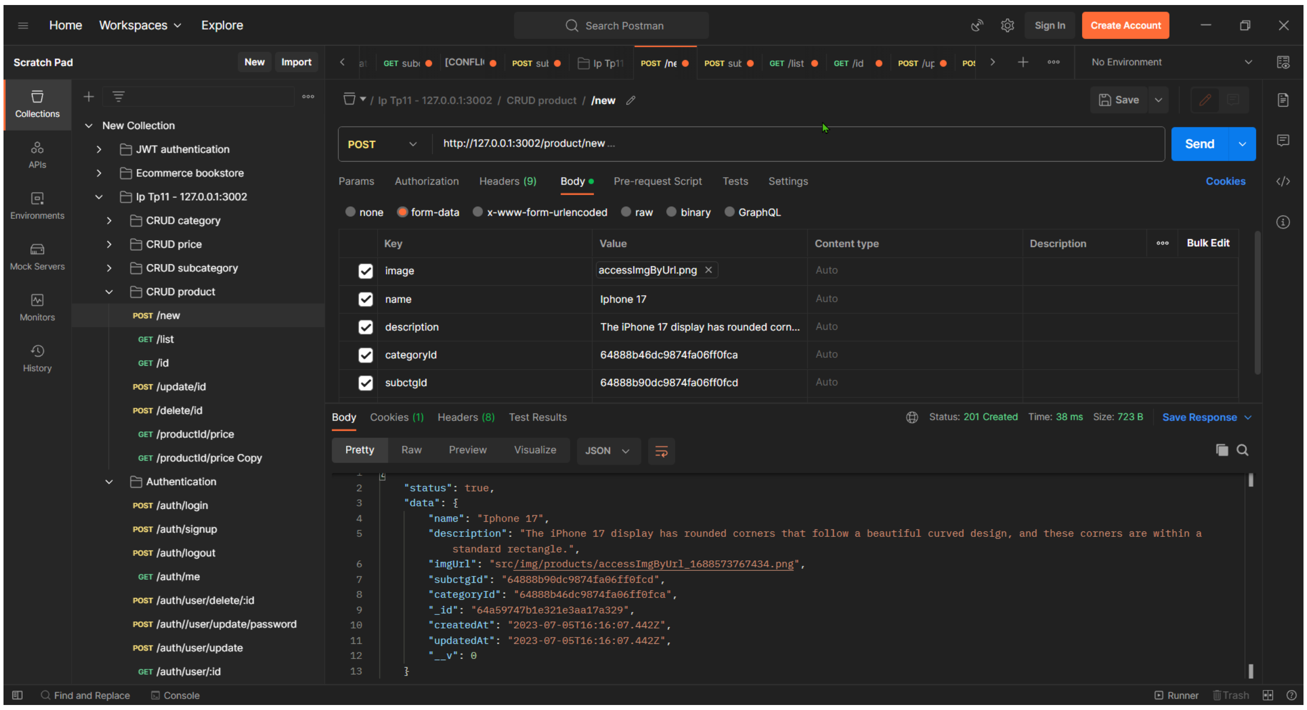


localhost:3002/static/get/src/img/products/fileMng_1688573356566.png

```typescript
import { Injectable } from '@nestjs/common';
import * as fs from 'fs';
import { extname, parse } from 'path';

import { createWriteStream, existsSync, mkdirSync } from 'fs';


@Injectable()
export class FileMngService {

  saveImage(file: any, path: string) {
    if (!file) {
      throw new Error('No file uploaded');
    }
    let { originalname, buffer } = file;
    if (!buffer || buffer.length === 0) {
      throw new Error('Empty file buffer');
    }
    const fileName = parse(originalname).name.replace(/\s/g, '/') + "_" + Date.now() + extname(file.originalname);
    var filePath = `${path}/${fileName}`;
    if (!existsSync(`./${path}`)) {
      mkdirSync(`./${path}`);
    }
    const fileStream = createWriteStream(filePath);
    fileStream.on('error', (error) => {
      throw error;
    });
    fileStream.write(buffer);
    fileStream.end();
    return filePath;
  }

  // delete file
  deleteFile(path: string) {
    try {
      fs.unlinkSync(path);
      //file removed
    } catch(err) {
      console.error(err)
    }
  }

  // replace file
  deleteAndAdd(olfFilePath: string, newFile: any, newPath: string) {
    this.deleteFile(olfFilePath);
    return this.saveImage(newFile, newPath);
  }
}
```

Create product with image

Now you can add product with image by using **form-data**.



POST http://127.0.0.1:3002/product/new

Ip Tp11 - 127.0.0.1:3002 / CRUD product / **/new**

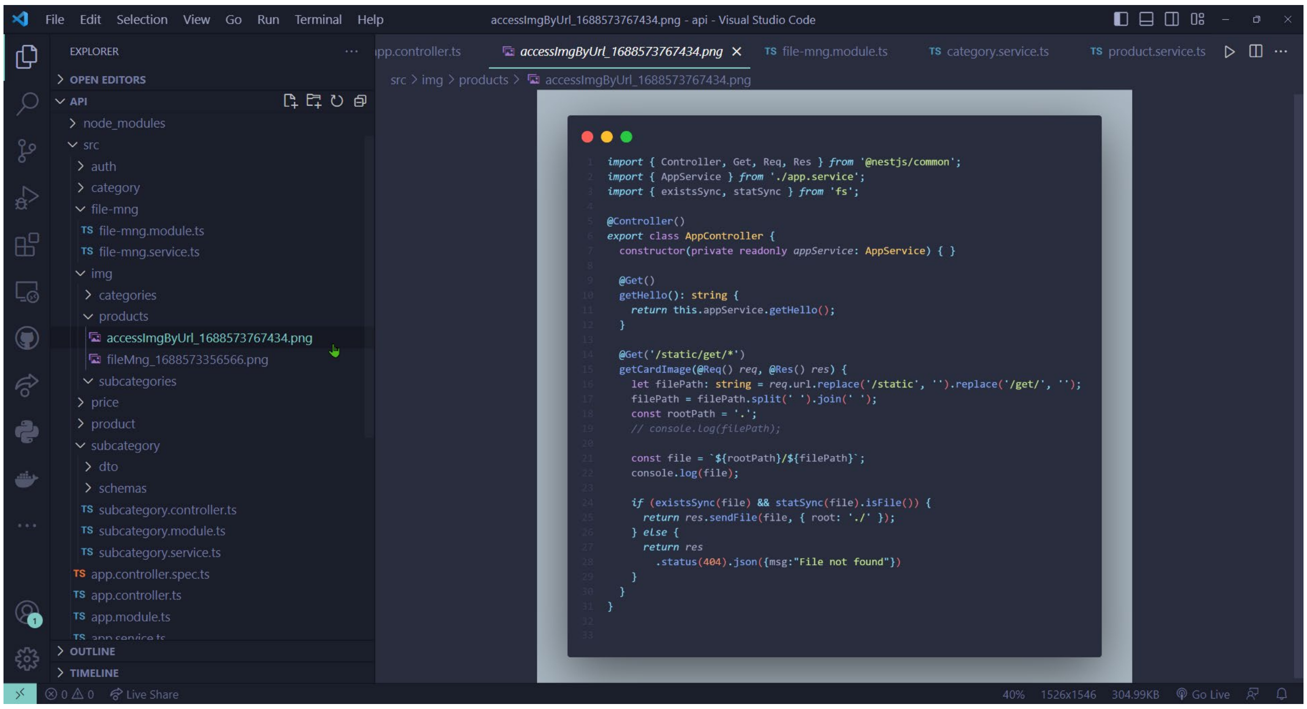
Body: form-data

	Key	Value	Content type	Description
✓	image	accessImgByUrl.png	Auto	
✓	name	Iphone 17	Auto	
✓	description	The iPhone 17 display has rounded corn...	Auto	
✓	categoryId	64888b46dc9874fa06ff0fca	Auto	
✓	subctgId	64888b90dc9874fa06ff0fcd	Auto	

Status: 201 Created Time: 38 ms Size: 723 B

```json
{
    "status": true,
    "data": {
        "name": "Iphone 17",
        "description": "The iPhone 17 display has rounded corners that follow a beautiful curved design, and these corners are within a
            standard rectangle.",
        "imgUrl": "src/img/products/accessImgByUrl_1688573767434.png",
        "subctgId": "64888b90dc9874fa06ff0fcd",
        "categoryId": "64888b46dc9874fa06ff0fca",
        "_id": "64a59747b1e321e3aa17a329",
        "createdAt": "2023-07-05T16:16:07.442Z",
        "updatedAt": "2023-07-05T16:16:07.442Z",
        "__v": 0
    }
```

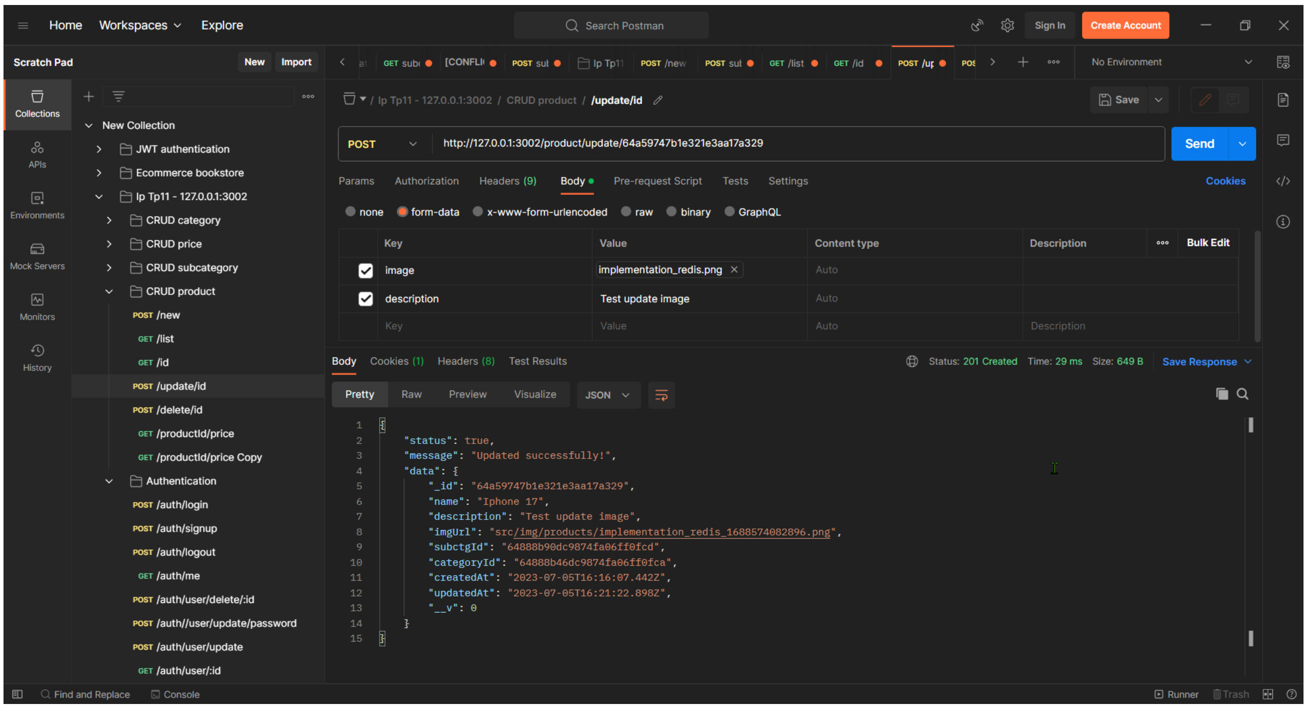
And this is the file we have stored in the folder **src/img/products**



```
import { Controller, Get, Req, Res } from '@nestjs/common';
import { AppService } from './app.service';
import { existsSync, statSync } from 'fs';

@Controller()
export class AppController {
  constructor(private readonly appService: AppService) { }

  @Get()
  getHello(): string {
    return this.appService.getHello();
  }

  @Get('/static/get/*')
  getCardImage(@Req() req, @Res() res) {
    let filePath: string = req.url.replace('/static', '').replace('/get/', '');
    filePath = filePath.split(' ').join(' ');
    const rootPath = '.';
    // console.log(filePath);

    const file = `${rootPath}/${filePath}`;
    console.log(file);

    if (existsSync(file) && statSync(file).isFile()) {
      return res.sendFile(file, { root: './' });
    } else {
      return res
        .status(404).json({msg:"File not found"})
    }
  }
}
```

Update product with image

You can update the image of the product too, it will delete the old image in the folder, then add the new one in the same directory.



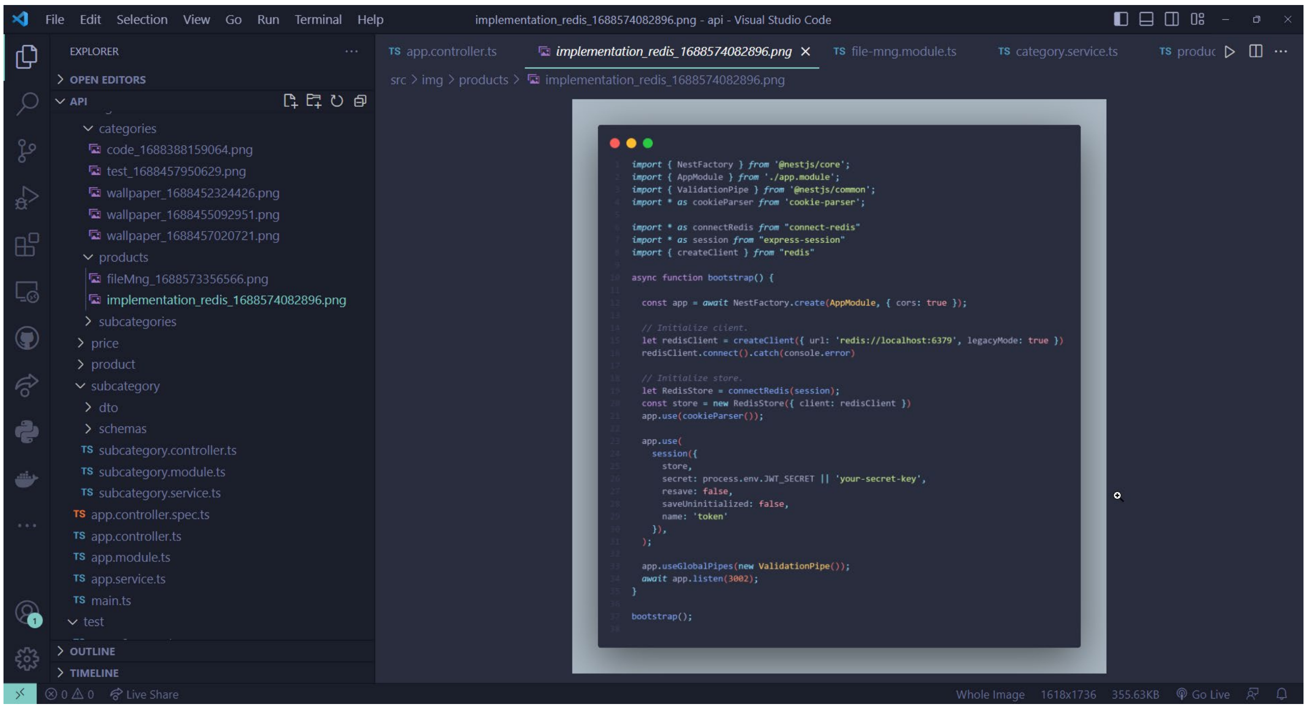
POST http://127.0.0.1:3002/product/update/64a59747b1e321e3aa17a329

Key	Value	Content type	Description
✓ image	implementation_redis.png	Auto	
✓ description	Test update image	Auto	

Status: 201 Created Time: 29 ms Size: 649 B

```
{
    "status": true,
    "message": "Updated successfully!",
    "data": {
        "_id": "64a59747b1e321e3aa17a329",
        "name": "Iphone 17",
        "description": "Test update image",
        "imgUrl": "src/img/products/implementation_redis_1688574082896.png",
        "subctgId": "64888b90dc9874fa06ff0fcd",
        "categoryId": "64888b46dc9874fa06ff0fca",
        "createdAt": "2023-07-05T16:16:07.442Z",
        "updatedAt": "2023-07-05T16:21:22.898Z",
        "__v": 0
    }
}
```

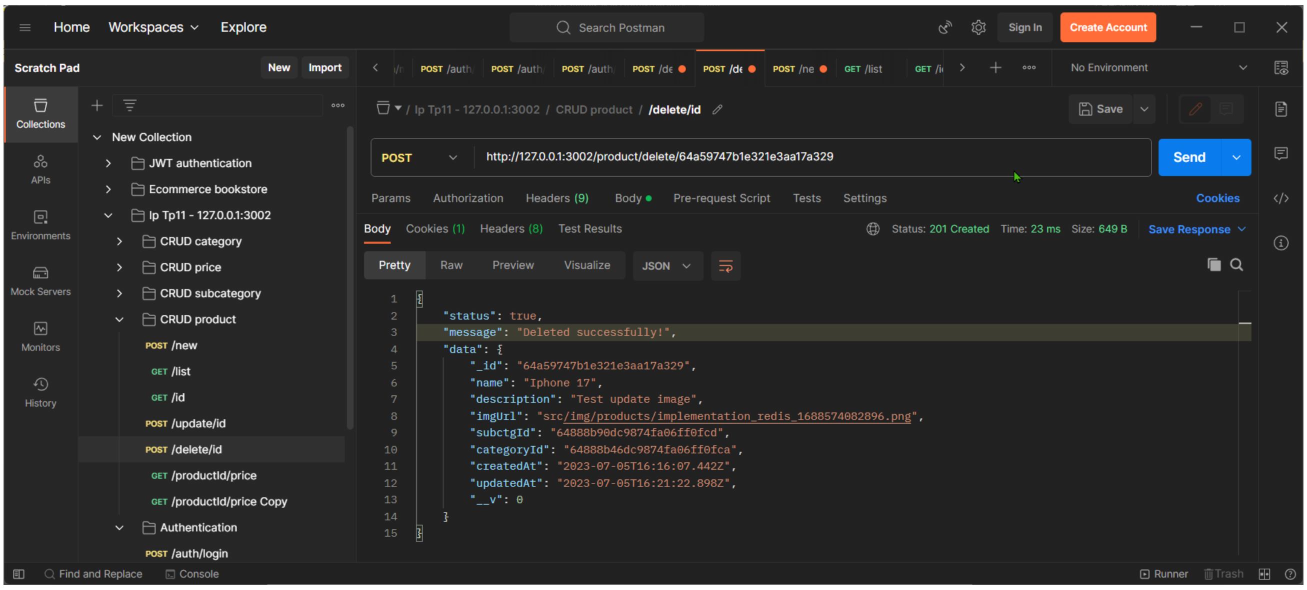
This is the changing of image that store in the directory **src/img/products**



```
import { NestFactory } from '@nestjs/core';
import { AppModule } from './app.module';
import { ValidationPipe } from '@nestjs/common';
import * as cookieParser from 'cookie-parser';

import * as connectRedis from "connect-redis"
import * as session from "express-session"
import { createClient } from "redis"

async function bootstrap() {

  const app = await NestFactory.create(AppModule, { cors: true });

  // Initialize client.
  let redisClient = createClient({ url: 'redis://localhost:6379', legacyMode: true })
  redisClient.connect().catch(console.error)

  // Initialize store.
  let RedisStore = connectRedis(session);
  const store = new RedisStore({ client: redisClient })
  app.use(cookieParser());

  app.use(
    session({
      store,
      secret: process.env.JWT_SECRET || 'your-secret-key',
      resave: false,
      saveUninitialized: false,
      name: 'token'
    }),
  );

  app.useGlobalPipes(new ValidationPipe());
  await app.listen(3002);
}

bootstrap();
```

Delete product

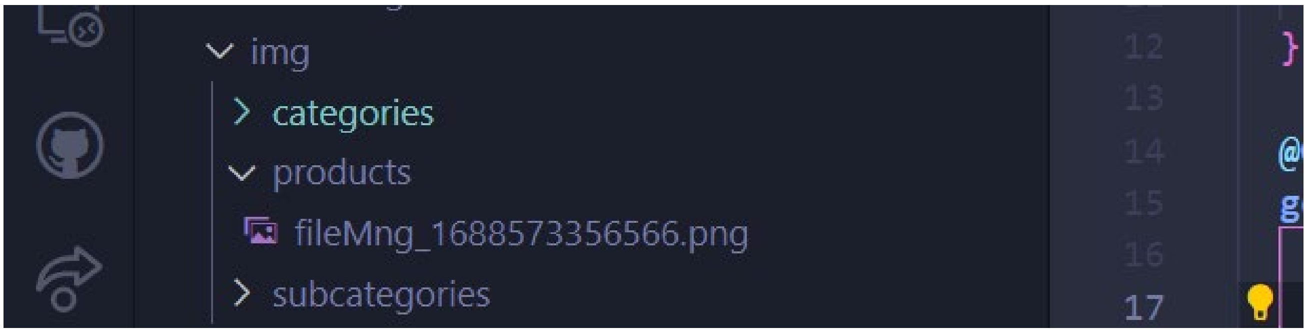
When you delete product, it will delete in the database then delete the image in the directory.



POST http://127.0.0.1:3002/product/delete/64a59747b1e321e3aa17a329

Status: 201 Created Time: 23 ms Size: 649 B

```
{
    "status": true,
    "message": "Deleted successfully!",
    "data": {
        "_id": "64a59747b1e321e3aa17a329",
        "name": "Iphone 17",
        "description": "Test update image",
        "imgUrl": "src/img/products/implementation_redis_1688574082896.png",
        "subctgId": "64888b90dc9874fa06ff0fcd",
        "categoryId": "64888b46dc9874fa06ff0fca",
        "createdAt": "2023-07-05T16:16:07.442Z",
        "updatedAt": "2023-07-05T16:21:22.898Z",
        "__v": 0
    }
}
```



Noted: for the **category** and **subcategory**, I have implemented the same as **product**.